XHTML™-Print

Draft 0.951
March 7, 2002

™ XHTML is a trademark of the World Wide Web Consortium.

Table of Contents

1 OVERVIEW

This section is informative.

This document specifies a simple XHTML based data stream suitable for printing as well as display. It is based on the W3C's XHTML Basic with the addition of cascading style sheets (CSS). Its targeted usage is for printing in environments where it is not feasible or desirable to install a printer-specific driver and where some variability in the formatting of the output is acceptable. Throughout this document this data stream is called "XHTML-Print."

XHTML-Print is designed to be appropriate for low-cost printers that may not have a full-page buffer and that generally print from top-to-bottom and left-to-right with the paper in a portrait orientation. For other printers (i.e., those that print in another direction or orientation) a full-page buffer may be required.

XHTML-Print is not appropriate when strict layout consistency and repeatability across printers are required. The design objective of XHTML-Print is to provide a relatively simple, broadly supportable page description format where content preservation and reproduction are the goal, i.e. "Content is King." Traditional printer page description formats such as PostScript or PCL are more suitable when strict layout control is required. XHTML-Print does not utilize bi-directional communications with the printer either for capabilities or status inquiries.

This document creates a set of conformance criteria for XHTML-Print. It includes style sheet constructs drawn from CSS2 and proposed for CSS3 to provide a strong basis for rich printing results without a detailed understanding of each individual printer's characteristics.

It also defines an extension set (or sets) that provide stronger layout control for the printing of mixed text and images, tables and image collections.

2 REFERENCES

This section is informative.

The following definitions and references are used throughout the document.

1. XHTML™ 1.0: The Extensible HyperText Markup Language. A reformulation of HTML 4.0 as an XML application. See http://www.w3.org/TR/xhtml1
2. XHTML™ Basic: A subset of XHTML 1.0 that includes a reduced set of functions. See http://www.w3.org/TR/xhtml-basic
3. Modularization of XHTML™: A document that describes an abstract modularization of XHTML and an implementation of the abstraction using XML Document Type Definitions (DTDs). This modularization provides a means for subsetting and extending XHTML, a feature needed for extending XHTML's reach onto emerging platforms. See http://www.w3.org/TR/xhtml-modularization
4. XML 1.0: The Extensible Markup Language (XML) is a subset of SGML that is to be served, received, and processed on the Web in the way that is not possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML. See http://www.w3.org/TR/REC-xml
5. CSS1: Cascading Style Sheets, level 1 is a simple style sheet mechanism that allows authors and readers to attach style (e.g. fonts, colors and spacing) to HTML documents. See http://www.w3.org/TR/REC-CSS1
6. CSS2: Cascading Style Sheets, level 2 build on CSS1 and, with very few exceptions, all valid CSS1 style sheets are valid CSS2 style sheets. CSS2 supports media-specific style sheets so that authors may tailor the presentation of their documents to visual browsers, aural devices, printers, braille devices, handheld devices, etc. CSS2 also adds content positioning, downloadable fonts, table layout, features for internationalization, automatic counters and numbering, and some properties related to user interface. See http://www.w3.org/TR/REC-CSS2/
7. "Namespaces in XML", T. Bray, D. Hollander, A. Layman, 14 January 1999. XML namespaces provide a simple method for qualifying names used in XML documents by associating them with namespaces identified by URI. Available at: http://www.w3.org/TR/REC-xml-names.
8. "Simple Object Access Protocol (SOAP) 1.1" SOAP is a lightweight XML-based protocol for exchange of information in a decentralized, distributed environment. It is a submission to the W3C and is available at http://www.w3.org/TR/SOAP.
9. "Introduction to CSS3" is an introduction and roadmap of the work in progress on CSS level 3. It is available from: http://www.w3.org/TR/css3-roadmap/
10. "RFC 2045 - Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed, N. Borenstein, Section 6.8. "Base64 Content-Transfer-Encoding" is of particular interest It is available at http://www.ietf.org/rfc/rfc2045.txt
11. "XML Schema Part 2: Datatypes W3C Recommendation 02 May 2001", P. Biron, A Malhotra. It is available at http://www.w3.org/TR/xmlschema-2/
12. "JPEG File Interchange Format, version 1.02, September 1, 1992", C-Cube Microsystems. Available from ftp://ftp.uu.net/graphics/jpeg/jfif.ps.gz or ftp://ftp.uu.net/graphics/jpeg/jfif.txt.gz

13. “RFC2119 - Key words for use in RFCs to Indicate Requirement Levels”, S. Bradner. It is available from http://www.ietf.org/rfc/rfc2119.txt
14. “RFC2387 - The MIME Multipart/Related Content-type”, E. Levinson. It is available from http://www.ietf.org/rfc/rfc2387.txt
15. “The MIME Application/Multiplexed Content-type”, Robert Herriot. It is currently available from http://search.ietf.org/internet-drafts/draft-herriot-application-multiplexed-04.txt (Subsequent versions will be available from the same location with the “04” incremented.)
16. “CCITT Recommendation T.81 | ISO/IEC 10918-1, Digital Compression and Coding of Continuous-tone Still Images: Requirements and Guidelines", ISO. Available from http://www.iso.org/iso/en/StandardsQueryFormHandler.StandardsQueryFormHandler
17. “JEIDA-49-1998 Digital still camera image file format standard(exif)”, Japan Electronics and Information Technology Industries Association (JEITA). Available from http://tsc.jeita.or.jp/
18. “Paged Media Properties for CSS3”, Robert Stevahn. Available from http://www.w3.org/TR/1999/WD-css3-page-19990928
19. “RFC3236 - The 'application/xhtml+xml' Media Type”, M. Baker. It is available from http://www.ietf.org/rfc/rfc3236.txt

3 TERMINOLOGY

This section is normative.

The keywords “MUST”, “SHALL”, “MUST NOT”, “SHALL NOT”, “REQUIRED”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” when used in this document are to be interpreted as described in RFC 2119. However, for readability, these words do not appear in all uppercase letters in this specification.

4 XHTML-PRINT ELEMENTS

This section is normative.

4.1 Elements Required by XHTML-Print

The World Wide Web Consortium has defined a subset of XHTML 1.0 that is appropriate for small format devices such as PDAs and mobile telephones. The subset is called XHTML-Basic and is useful to be examined as a starting point for the definition of XHTML-Print. XHTML-Print is a proper superset of XHTML Basic in the set of elements required. As listed below, the "Text Extension Modules," "Style Sheet Module," and "Style Attribute Module" are added to the modules required by XHTML Basic.

The Modularization of XHTML [3] is a decomposition of XHTML and by reference HTML 4.0 into a collection of abstract modules that provide specific types of functionality. XHTML-Print is defined, in part, by inclusion of a set of these modules. As such, the non-CSS portion of XHTML-Print includes the following modules:

1. Core Modules
   - Structure Module
     body, head, html, title
   - Text Module
     abbr, acronym, address, blockquote, br, cite, code, dfn, div, em, h1, h2, h3, h4, h5, h6, kbd, p, pre, q, samp, span, strong, var
   - Hypertext Module
     a
   - List Module
     dl, dt, dd, ol, ul, li
2. Text Extension Modules*
   - Presentation
     b, big, hr, i, small, sub, sup, tt
3. Basic Table Modules
   - Basic Table Module
     caption, table, td, th, tr
4. Image Module
   img
5. Basic Forms Modules
   - Basic Forms Module
     form, input, label, select, option, textarea
6. Metainformation Module
   meta
7. Style Sheet Module*
   style
8. Style Attribute Module*
9. Link Module
   link

10. Base Module
    base
11. Object Module
    object
    param

Note: Modules above marked by "*" are not a part of XHTML Basic but are required for XHTML-Print.

4.2 Restrictions on XHTML by XHTML-Print

XHTML-Print restricts some usage of common XHTML 1.0 elements in the same way that XHTML Basic does:

- Nesting of Tables is NOT supported
- Frames are NOT supported
- Script and Event Handling are NOT supported

4.3 CSS Conformance

See section 6.3.2 "Printer Requirements" for CSS conformance requirements for XHTML-Print conforming implementations.

5 APPLICATION OF XHTML/CSS FOR XHTML-PRINT

This section is normative.

XHTML-Print inherits all the structure, encoding and other basic infrastructure specified by XHTML. The following sections describe and clarify the application and usage restrictions of XHTML-Print.

5.1 Recommended Attributes on the 'img' and 'object' Elements

Because many printers create the page in a serial manner from top to bottom, it is important for the printer to know the size of images before retrieving the image data itself. This information is then used to create portions of the page layout.

Therefore, the sender is strongly encouraged to include the height and width attributes either within the *'img'* or the '*object'* element, or within an associated style sheet rule. Printers may omit from the page images that do not include height and width attributes (see Section 6.3.1, item 4). These attributes may be expressed as percentages within the '*img'* or the *'object'* element, or may use the standard absolute or relative units within the CSS rule. Percentages are relative to the parent element and not the page width or printable area.

This document specifies only one mandatory image format: baseline JPEG as defined in [12]. See Appendix A for a description of JPEG decoder requirements. Printers are not required to support:

- Embedded thumbnails
- Rotation
- Progressive rendering

within the JFIF and EXIF files.

5.2 Style Sheets

Conforming XHTML-Print printers shall support both in-line and referenced style sheets within the *'style'* element or '*link'* element in the '*head'* element of a document. Conforming XHTML-Print printers shall also support the style attributes (i.e. in-line style) when used within other elements as defined by XHTML 1.1. Normal cascading rules apply. See section 6.3.1 for special cases when the style section includes special characters such as "&" and "<".

5.3 Page Breaks

Because of the differences in displaying in a browser versus a paged media device like a printer, the content provider may want to have control over the locations of page breaks. Conforming implementations shall support the CSS2 ***page-break-before, page-break-after, and page-break-inside*** properties as the means of providing this control.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//PWG//DTD XHTML-Print 1.0//EN"
     "http://www.xhtml-print.org/xhtml-print/xhtml-print10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <title> Inline Page Breaks </title>
   <style>
   .pagebreak { page-break-after: always }
   </style>
</head>
 <body>
<p class="pagebreak">Page 1 Text</p>
<p>Page 2 Text</p>
 </body>
</html>
```

Figure 1: A Page Break Example

If ***page-break-inside: avoid*** is specified for a long element and the printer is unable to buffer the entire element before committing it to paper, it should force a page break to occur before the long element and begin the element starting at the top of the next page. If the long element starts at the top of a page and exceeds the page length, the printer shall print as much as possible on the first page and then resume that element on the next and subsequent pages as required to preserve the content. A printer is neither required nor forbidden to perform scaling to fit the long element on a single page.

5.4 Page Size and Orientation

(The following is a summary and partial extraction from the CSS2 document)

Page size and orientation are controlled using the @page rules from CSS2. Specifically, the size property is applied to @page to set both size and orientation. The margin properties ('margin-top', 'margin-bottom', 'margin-left', 'margin-right', and 'margin') as defined by CSS2 also apply within the page context. Page size and orientation that is provided in the XHTML-Print datastream will override similar attributes contained within any commands and/or attributes provided by job-submission protocols.

Due to a printer's mechanical limitations, the actual printable area of the page is usually less than the page size. Care must therefore be taken when generating XHTML-Print mark-up to allow for this reduced area when using certain features including but not limited to absolute positioning.

Results are printer-dependent when the CSS size specified does not match the media size being used.

5.4.1 Size Property

Usage:

@page {

```
    size: auto;  /* auto is the initial value and uses the default paper */
        }
```

‘size’

Value: <length>{1,2} | auto | portrait | landscape | inherit
Initial: auto
Applies to: the page context
Inherited: N/A
Percentages: N/A
Media: paged

This property specifies the size and orientation of a page box. The size of a page box may either be “absolute” (fixed size) or “relative” (scalable, i.e., fitting available sheet sizes). Relative page boxes allow printers to scale a document and make optimal use of the target size. Three values for the ‘size’ property create a relative page box:

- auto: The page box will be set to the size and orientation of the target sheet.
- landscape: Overrides the target’s orientation. The page box is the same size as the target, and the longer sides are horizontal.
- portrait: Overrides the target’s orientation. The page box is the same size as the target, and the shorter sides are horizontal.

5.4.2 Margin Properties

The margin properties are supported as specified in CSS2, clause 8.3. See also Table 1 on page 21.

5.4.3 Examples

In the following example, the outer edges of the page box will align with the target. The percentage value on the ‘margin’ property is relative to the target size so if the target sheet dimensions are 21.0cm x 29.7cm (i.e., A4), the margins are 2.10cm and 2.97cm.

```
@page {
  size: auto;   /* auto is the initial value */
  margin: 10%;
        }
```

Length values for the ‘size’ property create an absolute page box. If only one length value is specified, it sets both the width and height of the page box (i.e., the box is a square). Since the page box is the initial containing block, percentage values are not allowed for the ‘size’ property.

For example:

```
@page {
  size: 8.5in 11in;       /* width height */
        }
```

The above example sets the width of the page box to be 8.5in and the height to be 11in. The page box in this example requires a target sheet size of 8.5"x11" or larger. Since the width is smaller than the height, the orientation is portrait. Printers may allow users to control, via a means external to XHTML-Print and CSS, the transfer of the page box to the sheet (e.g., rotating an absolute page box that's being printed).

5.4.4 Rendering Page Boxes that do not Fit a Target Sheet

If a page box does not fit the target sheet dimensions, the printer may choose (in order of preference) to:

- Rotate the page box 90° if this will make the page box fit.
- Scale the page to fit the target. (There is no requirement to maintain the aspect ratio of the page or of any elements on the page when scaling; however, preservation of the aspect ratio is preferred.)
- Reformat the page (including "spilling" onto another sheet)
- Clip (least preferred)

The printer may consult the user before performing these operations. Lacking "access" to the user, it may simply make a decision on its own.

5.4.5 Positioning the Page Box on the Sheet

When the page box is smaller than the target size, the printer is free to place the page box anywhere on the sheet. However, it is recommended that the page box be centered on the sheet since this will align double-sided pages and avoid accidental loss of information that is printed near the edge of the sheet.

5.5 Running Headers and Footers

A means is needed to create a *running-header* and a *running-footer* on the printed page. Current work in progress by the W3C on paged media defines a very robust method for adding margin boxes to the top, bottom, left and right of the page. (See [18].) A reduced set from the CSS3 proposal is employed by XHTML-Print, using top and bottom margin boxes via the @page rules method.

Utilizing the terminology of CSS2 and CSS3, a 'margin box' is defined in conjunction with the 'page box' and 'page area' (as shown in Figure 2) to create an area into which *running-header* and *running-footer* text can be inserted.

CSS3 proposes the ability to left-align, right-align and center the text horizontally as well as methods to top-align, bottom-align and center the text vertically within the margin boxes. For XHTML-Print conforming implementations vertical controls are not required to be supported. Instead, for XHTML-Print conforming implementations, the *running-header* text may be top aligned in the margin box and the *running-footer* text may be bottom aligned in the margin box.

CSS3 proposes methods for the printing device to automatically include:

- page number
- total pages in the document
- date
- time
- file name

into the *running-header* and *running-footer*. XHTML-Print conforming implementations are only required to support inserting a page number. If required, the sending appliance must provide the other information within the text string to be printed in the margin box.

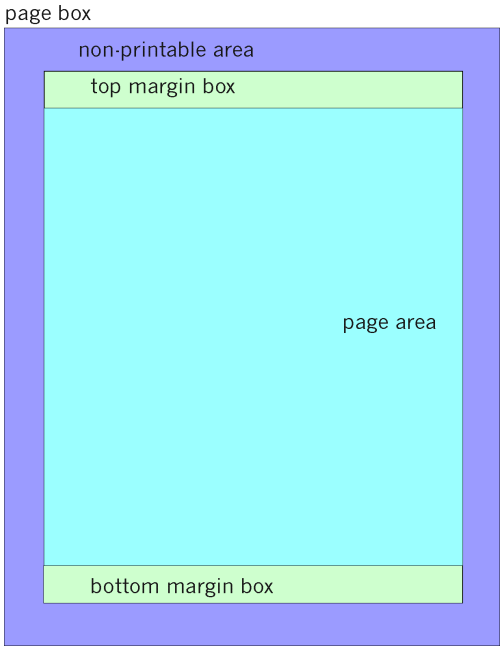

Figure 2

The following are sample XHTML/CSS fragments used to create *running-headers* and *running-footers.*

```
<style>
@page {
       @top{font-family: Helvetica, Arial, sans-serif;
            font-size: 150%;
            font-weight: bolder;
            text-align: left;
            content: "XHTML-Print: A Proposal --- August 25, 2000";
           }
       }
</style>
```

The above example creates a running header that is left aligned at 150% of normal font size and bold in Helvetica, Arial or the default san-serif font whichever is available.

```
<style>
@page {
       counter-increment: pages;
       @bottom{font-family: Times, Palatino, serif;
               font-size: 80%;
               font-weight: normal;
               text-align: center;
               content: "Page " counter(pages);
              }
       }
</style>
```

The above example creates a running footer such as "Page 14" centered on the page in a font 80% of normal size in Times, Palatino or the default serif font whichever is available. Note that since the counter named "pages" is both incremented and used by the @page rule, it will first be incremented and then used; so the header on the first page will be "Page 1".

5.6 Image Data

In traditional web-based applications of XHTML, image data is contained in a separate file on a web server that the user agent retrieves.

However, there are circumstances where it is desirable to include the image data along with the rest of the print data. Some low cost, resource constrained clients may want to include images in their print output but cannot afford to include a server. Some print applications may require that all the print data can be encapsulated in a single file for transportability, avoiding firewall issues, etc.

See Appendix B for discussion of the method that shall be used to collect both XHTML-Print and associated image data into a single file or data stream.

5.7 Side-by-Side Images

Low-cost printers today often have very little memory into which page data can be stored before being printed. As such, they must build and print the page in swaths on the fly from the top of the page to the bottom. To enable the use of XHTML-Print in these low cost printers, some restrictions on the order of images contained in the XHTML-Print data stream must be added.

1. If two or more images will be even partially side-by-side on the printed page they should be included by reference (<img src="http://10.10.10.2/images/logo.jpg"> or <object data="http://10.10.10.2/images/logo.jpg">) rather than included inline. (See Appendix B). This allows the printer to get chunks of the image, as it needs it, as it prints down the page.

2. An XHTML-Print conforming printer lacking sufficient buffer space to hold multiple side-by-side images may choose to reformat the layout of the page to preserve content. Printers shall attempt to preserve content when encountering side-by-side images that may be impossible to print as specified within the XHTML-Print. Discarding the second and subsequent of the side-by-side images should be avoided unless preservation of content is best achieved by doing so. Other than attempting to best preserve content, this specification does not mandate any specific behavior when encountering this situation. Clients providing images inline should order them from left-to-right top-to-bottom unless the print direction is known to be otherwise.

6 CONFORMANCE

This section is normative.

6.1 XHTML-Print Document Type Conformance

A conforming XHTML-Print document is a document that requires only the facilities described as mandatory in this specification. Such a document must meet all of the following criteria:

1. The document must validate against the DTD found in Appendix C and conform to the constraints expressed in Section 4.2.
2. The root element of the document must be <html>.
3. The name of the default namespace on the root element must be the XHTML namespace name, http://www.w3.org/1999/xhtml.
4. There must be a DOCTYPE declaration in the document prior to the root element. If present, the public identifier included in the DOCTYPE declaration must reference the DTD found in Appendix C using its Formal Public Identifier. The system identifier may be modified appropriately.

<!DOCTYPE html PUBLIC "-//PWG//DTD XHTML-Print 1.0//EN"
"http://www.xhtml-print.org/xhtml-print/xhtml-print10.dtd">

The MIME type used to refer to a conforming XHTML-Print document shall be "application/vnd.pwg-xhtml-print+xml"

6.2 Client Conformance

1. Clients shall produce a well-formed XHTML-Print document as defined in [1] and in Section 6.1.
2. Beyond number 1 above, clients are not required to use more of the XHTML-Print elements or Style Sheet attributes than necessary to get the desired output.

6.3 Printer Conformance

6.3.1 Formatting/Rendering Rules

1. In order to be consistent with the XML 1.0 Recommendation [4], the printer must parse and evaluate an XHTML-Print document to determine if the document is well formed. If the printer claims to be a validating printer, it must also validate documents against their referenced DTDs according to XML. Validation is not required to claim conformance to this standard. A printer may "flush" or otherwise reject a non-conforming XHTML-Print document.
2. When the printer claims to support facilities defined within this specification or required by this specification through normative reference, it must do so in ways consistent with the facilities' definition.

3. When a printer processes an XHTML-Print document as generic XML, it shall only recognize attributes of type ID (e.g. the id attribute on most XHTML elements) as fragment identifiers.
4. Images:
    - If a printer encounters an image in a format it does not support, it shall render any alternate content provided, and may reserve the space specified by the height and width attributes by optionally drawing a box around this space of the size specified for the image.
    - If the image format is not supported or the height and width attributes are absent and no alternate content is provided, the image may be omitted and no space reserved.
    - If the image format is supported and the height and width attributes were omitted, the printer may choose to omit the image from the page.
5. If a printer encounters an element it does not recognize, it must continue to process the children of that element. If the content is text, the text should be presented to the user.. Printers may chose not to render content within elements defined by XHTML, HTML or deprecated from HTML which is obviously not intended to be rendered, e.g. <script>.
6. If a printer encounters an attribute it does not recognize, it must ignore the entire attribute specification (i.e., the attribute and its value).
7. If a printer encounters an attribute value it doesn't recognize, it must use the default attribute value.
8. If a printer encounters an entity reference, e.g. "&#X007;", (other than one of the predefined entities) for which the printer has processed no declaration (which could happen if the declaration is in the external subset which the printer hasn't read), the entity reference should be rendered as the characters (starting with the ampersand and ending with the semi-colon) that make up the entity reference.
9. When rendering content, printers that encounter characters or character entity references that are recognized but not renderable should display the document in such a way that it is obvious to the user that normal rendering has not taken place.
10. Whitespace is handled according to the following rules. The following characters are defined in XML as whitespace characters: Space ( ) Tab (	) Carriage return (
) Line feed (
). The XHTML-Print printer in addition must treat the following characters as whitespace: Form feed () and Zero-width space (). The XML processor normalizes different systems' line end codes into one single Line feed character that is passed up to the application.

The printer must process whitespace characters in the data received from the XML processor as follows:

- All whitespace surrounding block elements should be removed.
- Comments are removed entirely and do not affect whitespace handling. One whitespace character on either side of a comment is treated as two whitespace characters.
- Leading and trailing whitespace inside a block element must be removed.
- When the '`xml:space`' attribute is set to '`preserve`', whitespace characters must be preserved and consequently Line feed characters within a block must not be converted.
- When the '`xml:space`' attribute is not set to '`preserve`', then:

- Leading and trailing whitespace inside a block element must be removed.
- Line feed characters must be converted into one of the following characters: a Space character, a Zero-width space character (), or no character (i.e. removed). The choice of the resulting character is printer dependent and is conditioned by the script property of the characters preceding and following the Line feed character.
- A sequence of whitespace characters without any Line feed characters must be reduced to a single Space character.
- A sequence of whitespace characters with one or more Line feed characters must be reduced in the same way as a single Line feed character.

Whitespace in attribute values is processed according to [XML].

Note (*informative*): In determining how to convert a Line feed character, a printer should consider the following cases, whereby the script of characters on either side of the Line feed determines the choice of the replacement. Characters of common script (such as punctuation) are treated as the same as the script on the other side:

1. If the characters preceding and following the Line feed character belong to a script in which the Space character is used as a word separator, the Line feed character should be converted into a Space character. Examples of such scripts are Latin, Greek, and Cyrillic.
2. If the characters preceding and following the Line feed character belong to an ideographic-based script or writing system in which there is no word separator, the Line feed should be converted into no character. Examples of such scripts or writing systems are Chinese, Japanese.
3. If the characters preceding and following the Line feed character belong to a non-ideographic-based script in which there is no word separator, the Line feed should be converted into a Zero-width space character () or no character. Examples of such scripts are Thai, Khmer.

- If none of the conditions in (1) through (3) are true, the Line feed character should be converted into a Space character.

6.3.2 Printer Requirements

- A conforming printer shall support all XHTML Modules listed in Section 4.1.
- A conforming printer shall print a static version of a form using default and selected values as specified in the form.
- A conforming printer shall identify this datastream by the exact string: “XHTML-Print” (without the quotation marks) in all service discovery records and protocols, device identification records and protocols, and in other cases where a list of supported datastreams is to be presented by the printer. Where such datastreams are identified by a MIME media type, the string “application/vnd.pwg-xhtml-print+xml” shall be used.

- A conforming printer shall support the CSS constructs and associated values as indicated in Section 6.3.2.1.

6.3.2.1 CSS Conformance Requirements

A conforming printer shall support the following CSS constructs and associated values; support for other values and other properties or constructs is optional:

CSS Constructs	Value
@media	print
@page	size, margin, margin-top, margin-right, margin-bottom, margin-left, :first, @top[3], @bottom[3], counter-increment, counter-reset
@bottom[3]	text-align, text-decoration, text-indent, color, font, font-family, font-size, font-style, font-weight, content
@top[3]	text-align, text-decoration, text-indent, color, font, font-family, font-size, font-style, font-weight, content
color	cname[1], #rrggbb, #rgb, rgb (r,g,b), rgb (r%,g%,b%), inherit
content	inherit, string, counter(name)
font	font-family, font-size, font-style, font-weight, line-height
font-family	font names and font-family names2 (It is recommended that XHTML-Print compliant printers minimally support “serif,” “san serif,” and “monospaced” font families.)
font-size[2]	xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, %, length (pt)
font-style	normal, italic[2], inherit
font-weight	all[2]
height	auto, inherit, length (in, mm), %
line-height	normal, inherit, in, mm, pt, ex units
list-style-position	inside, outside, inherit
list-style-type	disc, inherit, none, upper-alpha, lower-alpha, decimal
margin	auto, one, two, three and four values
margin-bottom	auto, inherit, %, length (in, mm)
margin-left	auto, inherit, %, length (in, mm)

margin-right	auto, inherit, %, length (in, mm)
margin-top	auto, inherit, %, length (in, mm)
page	string as defined by @page rule
page-break-after	auto, inherit, always
page-break-before	auto, inherit, always
page-break-inside	auto, inherit, avoid
size (used with @page)	auto, inherit, length, one and two values (in, mm), portrait
text-align	inherit, left, center
text-decoration	none, inherit, underline
text-indent	inherit, %, length (in, mm, ex)
white-space	normal, nowrap, pre, inherit
width	auto, inherit, %, length (mm, in)

Table 1 - CSS Conformance

[1] The 16 color names as defined in Section 4.3 of [3]

[2] The supported values should be appropriate to the fonts available to the printer.

[3] See [18] for a description of this functionality.

Printers are not required to support pixel length units. However, if they do so, they should conform with Section 4.3.2 of [6]; *i.e.,* for printers whose output is expected to be read at a little less than arm's length, there are about 121 pixels per inch.

6.4 Enhanced Layout Extension

To further support print applications requiring more exacting page layout (*e.g.,* photo album pages), the following additional style sheet properties and image format are required to be supported in an optional, discoverable (via some means outside the scope of this document) Enhanced Layout Extension. If support for this extension is indicated, all of the following must be supported:

- All CSS constructs listed in Table 1 plus all CSS constructs listed in Table 2.

- Image File Format
    - See Section A.3 for a description of additional JPEG decoding requirements for this extension.

CSS Constructs for Enhanced Layout	Value
@import	uri (<string>), <string>, print
border	border-width, border-color, border-style
border-bottom	border-width, border-color, border-style
border-bottom-color	color values as in Table 1
border-bottom-style	none, solid
border-bottom-width	inherit, medium, thick, thin, length (in, mm, ex, %)
border-color	one, two, three, and four color values; see “color” in Table 1 for color formats to be supported
border-left	border-width, border-color, border-style
border-left-color	color values as in Table 1
border-left-style	none, solid
border-left-width	inherit, medium, thick, thin, length (in, mm, ex, %)
border-right	border-width, border-style, border-color
border-right-color	color values as in Table 1
border-right-style	none, solid
border-right-width	inherit, medium, thick, thin, length (in, mm, ex, %)
border-spacing	inherit, length, one and two values (in, mm, ex,)
border-style	one, two, three, and four values (none, solid)
border-top	border-width, border-style, border-color
border-top-color	color values as in Table 1
border-top-style	none, solid
border-top-width	inherit, medium, thick, thin, length (in, mm, ex, %)
border-width	one, two, three, and four values
bottom	auto, inherit, length (in, mm, ex, %)
clear	none, left, right, both, inherit
clip	auto, inherit, rect (ltop, lright, lbot, lleft) in in, mm
display	inherit, block, inline, list-item, none
float	none, left, right, inherit
left	auto, inherit, length (in, mm, ex, %)
overflow	inherit, visible, hidden

padding	one, two, three, or four width values (in, mm, ex, %)
padding-bottom	inherit, width (in, mm, ex, %)
padding-left	inherit, width (in, mm, ex, %)
padding-right	inherit, width (in, mm, ex, %)
padding-top	inherit, width (in, mm, ex, %)
position	inherit, static, absolute, relative
right	auto, inherit, length (in, mm, ex, %)
table-layout	auto, fixed, inherit
top	auto, inherit, length (in, mm, ex, %)

Table 2 Additional CSS Constructs Required for Advanced Layout Extension

The following is an informative example using absolute positioning with image data:

```
<style>
.picture1 {
                position: absolute;
                top: 25mm;
                left: 25mm;
                padding-top: 10mm;
                width: 30mm;
                height: 30mm;
                clip: rect(10mm, 30mm, 30mm, 0mm)
            }
</style>
…
<div class=“picture1”>
<img src=“some_file.jpg” />
</div>
```

7 AUTHORS

This section is informative.

Authors' Addresses:

Don Wright
Lexmark International
don@lexmark.com
+1 859-232-4808

Melinda Grant
Hewlett-Packard
melinda_grant@hp.com
+1 360-212-3544

Peter Zehler
Xerox
pzehler@crt.xerox.com
+1 716-265-8755

Jun Fujisawa
Canon
fujisawa.jun@canon.co.jp
+81 44-733-6111

Appendix A JPEG DECODER REQUIREMENTS

A.1 Introduction

This section is normative.

A.1.1 Intent

This appendix describes recommended behaviors for JPEG decoders in XHTML-Print devices. Behaviors for both minimal printers and photo printers are described. Many of the behaviors described in this document follow directly from language already present in the relevant JPEG standards, but are repeated here to emphasize their importance.

A.1.2 Objectives

The decoder behaviors described in this document are intended to minimize implementation complexity, while retaining maximum compatibility with existing JPEG files. In particular, these recommendations seek to ensure compatibility with both EXIF and baseline JFIF (i.e. the subset of JFIF files that use only baseline JPEG processes). Support for JPEG streams using non-baseline processes, such as arithmetic coding or progressive coding, is not mandated for XHTML-Print compliance.

A.2 Behaviors of Minimal Printers

This section describes behaviors of JPEG decoders for minimal XHTML-Print implementations.

A.2.1 JPEG Processes

A JPEG decoder for an XHTML-Print printer must support all baseline JPEG processes as defined in section 3.11 of [16], except for 2- and 4-component images. These processes include greyscale and 3-component images, 8-bit/component sample depth, Huffman entropy coding, 444, 422, 411, and 400 subsampling modes, and sequential (i.e. non-progressive) scan.

A.2.2 Handling of APPx Markers

Baseline decoders may ignore application-specific markers, such as the JFIF APP0 marker and the EXIF APP1/APP2 markers. This will cause all images to print in an un-rotated orientation, with image size as specified in the JPEG SOF marker if not overridden by XHTML-Print mark-up. A JPEG decoder for a minimal printer must not fail as a consequence of encountering an unsupported APPx marker (i.e. all such markers must be correctly parsed, even if they are ignored).

A.2.3 Color Management

This section describes a recommended color management approach for minimal XHTML-Print printers.

Greyscale Images

Sample values in a grayscale (single-component) JPEG image shall be converted to the sRGB color space by setting

$$R_{out} = G_{out} = B_{out} = Gray_{in}$$

Color Images

Sample values in 3-component JPEG images shall be interpreted as YCbCr samples, as would be obtained by applying the matrices described in ITU BT.601 to sRGB input data.

A.3 XHTML-Print Enhanced Layout Extension

This section describes behaviors of JPEG decoders for XHTML-Print devices that support the XHTML-Print Enhanced Layout Extension, an optional feature block. The behaviors described below should be interpreted as "in addition to" those described in Section 4.1 and Section 6.3 (the requirements for minimal XHTML-Print devices).

A.3.1 Handling of EXIF APP1 and APP2 Markers

A JPEG decoder for an XHTML-Print implementation which supports the Enhanced Layout Extension must decode the TIFF IFDs embedded in the EXIF APP1 and APP2 markers, as described in Section 2.6.4 of [17]. The following IFDs must be fully supported (i.e. they must not be ignored).

Tag Name	Field Name	Description
Orientation of Image	Orientation	Sets image orientation in 90-degree increments, and enables transposition.

Appendix B INLINE IMAGE DAT A

B.1 Introduction

This section is normative.

B.1.1 Intent

The intent of this appendix is to describe the method for including XHTML-Print and associated image data in a single data stream or file. Support for Inline Image Data is conditionally mandatory; i.e. any device supporting Inline Image Data must support this method. (See Section 5.6.) Mandating support for Inline Image Data is outside the scope of this document.
In addition to images, if separate style sheets are to be interleaved with the XHTML-Print data, the same method shall be used.

B.1.2 Objectives

- Minimize image data size
- No or minimal additional encoding / decoding of image data required
- Enable juxtaposition between image data and associated XHTML-Print content
- Many printers are unable to buffer significant amounts of page content data, so the image data must be printed more or less as it is received. This implies the image data must be sent near the related XHTML-Print content, so that layout and printing can occur without extensive data buffering.
- Minimize complexity
- Leverage existing standard capabilities

B.2 MIME type Application/Multiplexed

This section includes by reference the entirety of "The MIME Application/Multiplexed Content-type", Robert Herriot. See [15].

B.3 Using OBJECT for In-Line Image Data

An alternative method to include inline image data in XHTML-Print is via the '*object'* element and a forward reference. The '*declare'* attribute of the '*object'* element is used to define the object, but delay its processing. The '*id'* attribute is used to associate the forward reference with the image content, sent at the end of the XHTML-Print document. Because this method normally encodes the binary image data using base64 encoding, a significant increase in the size of the data transmitted will be experienced. This should be avoided over low speed connections.

```
<object declare="declare"
 height="20 mm" width="20 mm"
 type="image/jpeg"
 id="image_1" >
</object>

. . . .

<object id="image_1"
 data="  . . . ">
</object>
```

This method may be useful for very simple clients that cannot afford a server for image download or for some reason cannot utilize the Application/Multiplexed MIME type; however, it is not recommended for general use especially if the size of the printer's buffer is unknown.

Appendix C XHTML-PRINT DT D AND MODULES

This section is normative.

This section contains the pieces of the XHTML-Print DTD that are unique to XHTML-Print. The remaining entities and modules are as specified in reference [3].

The following should be used from Modularization of XHTML [3]:

1. xhtml-attribs-1.mod
2. xhtml-base-1.mod
3. xhtml-basic-form-1.mod
4. xhtml-basic-table-1.mod
5. xhtml-blkphras-1.mod
6. xhtml-blkpres-1.mod
7. xhtml-blkstruct-1.mod
8. xhtml-charent-1.mod
9. xhtml-datatypes-1.mod
10. xhtml-framework-1.mod
11. xhtml-hypertext-1.mod
12. xhtml-image-1.mod
13. xhtml-inlphras-1.mod
14. xhtml-inlpres-1.mod
15. xhtml-inlstruct-1.mod
16. xhtml-inlstyle-1.mod
17. xhtml-lat1.ent
18. xhtml-link-1.mod
19. xhtml-list-1.mod
20. xhtml-meta-1.mod
21. xhtml-notations-1.mod
22. xhtml-object-1.mod
23. xhtml-param-1.mod
24. xhtml-pres-1.mod
25. xhtml-qname-1.mod
26. xhtml-special.ent
27. xhtml-struct-1.mod
28. xhtml-style-1.mod
29. xhtml-symbol.ent
30. xhtml-text-1.mod

C.1 XHTML-Print 1.0 DTD

```
<!-- ....................................................................... -->
<!-- XHTML-Print 1.0 DTD ................................................... -->
```

```
<!-- file: xhtml-print10.dtd
-->

<!-- XHTML-Print 1.0 DTD

     This is XHTML-Print 1.0, a variant of XHTML Basic for printing.

     Copyright 2001 Lexmark International Inc., Hewlett-Packard Company,
     Xerox Corporation, and Canon Inc. All Rights Reserved.

     Permission to use, copy, modify and distribute the XHTML-Print DTD and
     its accompanying documentation for any purpose and without fee is hereby
     granted in perpetuity, provided that the above copyright notice and
     this paragraph appear in all copies.  The copyright holders make no
     representation about the suitability of the DTD for any purpose.

     It is provided "as is" without expressed or implied warranty.

        Author:   Jun Fujisawa <fujisawa.jun@canon.co.jp>
        Revision: $Id: xhtml-print10.dtd,v 1.2 2001/06/04 17:16:35 fujisawa Exp $

-->
<!-- This is the driver file for version 1.0 of the XHTML-Print DTD.

   This DTD is identified by the PUBLIC and SYSTEM identifiers:

        PUBLIC "-//PWG//DTD XHTML-Print 1.0//EN"
        SYSTEM "http://www.xhtml-print.org/xhtml-print/xhtml-print10.dtd"
-->
<!ENTITY % XHTML.version "-//PWG//DTD XHTML-Print 1.0//EN" >

<!-- Use this URI to identify the default namespace:

         "http://www.w3.org/1999/xhtml"
-->
<!ENTITY % NS.prefixed "IGNORE" >
<!ENTITY % XHTML.prefix "" >

<!-- Reserved for use with the XLink namespace:
-->
<!ENTITY % XLINK.xmlns "" >
<!ENTITY % XLINK.xmlns.attrib "" >

<!-- reserved for future use with document profiles -->
<!ENTITY % XHTML.profile "" >

<!-- Bidirectional Text features
     This feature-test entity is used to declare elements
     and attributes used for bidirectional text support.
-->
<!ENTITY % XHTML.bidi "IGNORE" >

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!ENTITY % xhtml-events.module "IGNORE" >
<!ENTITY % xhtml-bdo.module "%XHTML.bidi;" >

<!-- Style Attribute Module ............................ -->
<!ENTITY % xhtml-inlstyle.module "INCLUDE" >
<![%xhtml-inlstyle.module;[
<!ENTITY % xhtml-inlstyle.mod
     PUBLIC "-//W3C//ENTITIES XHTML Inline Style 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-inlstyle-1.mod" >
%xhtml-inlstyle.mod;]]>

<!-- Document Model Module ............................. -->
<!ENTITY % xhtml-model.mod
     PUBLIC "-//PWG//ENTITIES XHTML-Print 1.0 Document Model 1.0//EN"
            "xhtml-print10-model-1.mod" >

<!-- Modular Framework Module (required) ............... -->
```

```
<!ENTITY % xhtml-framework.mod
     PUBLIC "-//W3C//ENTITIES XHTML Modular Framework 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-framework-1.mod" >
%xhtml-framework.mod;

<!-- Text Module (required) ........................... -->
<!ENTITY % xhtml-text.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Text 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-text-1.mod" >
%xhtml-text.mod;

<!-- Hypertext Module (required) ...................... -->
<!ENTITY % xhtml-hypertext.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Hypertext 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-hypertext-1.mod" >
%xhtml-hypertext.mod;

<!-- Lists Module (required) .......................... -->
<!ENTITY % xhtml-list.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Lists 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-list-1.mod" >
%xhtml-list.mod;

<!-- :::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!-- Presentation Module .............................. -->
<!ENTITY % xhtml-pres.module "INCLUDE" >
<![%xhtml-pres.module;[
<!ENTITY % xhtml-pres.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Presentation 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-pres-1.mod" >
%xhtml-pres.mod;]]>

<!-- Image Module ..................................... -->
<!ENTITY % xhtml-image.module "INCLUDE" >
<![%xhtml-image.module;[
<!ENTITY % xhtml-image.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Images 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-image-1.mod" >
%xhtml-image.mod;]]>

<!-- Tables Module .................................... -->
<!ENTITY % xhtml-table.module "INCLUDE" >
<![%xhtml-table.module;[
<!ENTITY % xhtml-table.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Basic Tables 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-basic-table-1.mod" >
%xhtml-table.mod;]]>

<!-- Forms Module ..................................... -->
<!ENTITY % xhtml-form.module "INCLUDE" >
<![%xhtml-form.module;[
<!ENTITY % xhtml-form.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Basic Forms 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-basic-form-1.mod" >
%xhtml-form.mod;]]>

<!-- Style Sheet Module ............................... -->
<!ENTITY % xhtml-style.module "INCLUDE" >
<![%xhtml-style.module;[
<!ENTITY % xhtml-style.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Style Sheets 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-style-1.mod" >
%xhtml-style.mod;]]>

<!-- Link Module ...................................... -->
<!ENTITY % xhtml-link.module "INCLUDE" >
<![%xhtml-link.module;[
<!ENTITY % xhtml-link.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Link Element 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-link-1.mod" >
```

```
%xhtml-link.mod;]]>

<!-- Metainformation Module ........................... -->
<!ENTITY % xhtml-meta.module "INCLUDE" >
<![%xhtml-meta.module;[
<!ENTITY % xhtml-meta.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Metainformation 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-meta-1.mod" >
%xhtml-meta.mod;]]>

<!-- Base Module ...................................... -->
<!ENTITY % xhtml-base.module "INCLUDE" >
<![%xhtml-base.module;[
<!ENTITY % xhtml-base.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Base Element 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-base-1.mod" >
%xhtml-base.mod;]]>

<!-- Param Module ..................................... -->
<!ENTITY % xhtml-param.module "INCLUDE" >
<![%xhtml-param.module;[
<!ENTITY % xhtml-param.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Param Element 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-param-1.mod" >
%xhtml-param.mod;]]>

<!-- Object Module .................................... -->
<!ENTITY % xhtml-object.module "INCLUDE" >
<![%xhtml-object.module;[
<!ENTITY % xhtml-object.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Embedded Object 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-object-1.mod" >
%xhtml-object.mod;]]>

<!-- Structure Module (required) ...................... -->
<!ENTITY % xhtml-struct.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Document Structure 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-struct-1.mod" >
%xhtml-struct.mod;

<!-- end of XHTML-Print 1.0 DTD ........................................ -->
<!-- ................................................................... -->
```

C.2 XHTML-Print 1.0 Document Model Module

```
<!-- ................................................................... -->
<!-- XHTML-Print 1.0 Document Model Module ............................. -->
<!-- file: xhtml-print10-model-1.mod

     This is XHTML-Print 1.0, a variant of XHTML Basic for printing.
    Copyright 2001 Lexmark International Inc., Hewlett-Packard Company,
     Xerox Corporation, and Canon Inc. All Rights Reserved.
     Revision: $Id: xhtml-print10-model-1.mod,v 1.2 2001/06/04 17:16:35 fujisawa Exp $

   This DTD module is identified by the PUBLIC and SYSTEM identifiers:

        PUBLIC "-//PWG//ENTITIES XHTML-Print 1.0 Document Model 1.0//EN"
        SYSTEM "http://www.xhtml-print.org/xhtml-print/xhtml-print10-model-1.mod

     ................................................................... -->

<!-- XHTML-Print 1.0 Document Model

     This module describes the groupings of elements that make up
     common content models for XHTML-Print elements.
-->

<!-- Optional Elements in head ......................... -->
```

```
<!ENTITY % HeadOpts.mix
     "( %meta.qname; | %link.qname; | %object.qname; | %style.qname; )*" >

<!-- Miscellaneous Elements ......................... -->

<!ENTITY % Misc.class "" >

<!-- Inline Elements ................................ -->

<!ENTITY % InlStruct.class "%br.qname; | %span.qname;" >

<!ENTITY % InlPhras.class
     "| %em.qname; | %strong.qname; | %dfn.qname; | %code.qname;
      | %samp.qname; | %kbd.qname; | %var.qname; | %cite.qname;
      | %abbr.qname; | %acronym.qname; | %q.qname;" >

<!ENTITY % InlPres.class
     "| %tt.qname; | %i.qname; | %b.qname; | %big.qname;
      | %small.qname; | %sub.qname; | %sup.qname; " >

<!ENTITY % I18n.class "" >

<!ENTITY % Anchor.class "| %a.qname;" >

<!ENTITY % InlSpecial.class "| %img.qname; | %object.qname;" >

<!ENTITY % InlForm.class
     "| %input.qname; | %select.qname; | %textarea.qname;
      | %label.qname;"
>

<!ENTITY % Inline.extra "" >

<!ENTITY % Inline.class
     "%InlStruct.class;
      %InlPhras.class;
      %InlPres.class;
      %Anchor.class;
      %InlSpecial.class;
      %InlForm.class;
      %Inline.extra;"
>

<!ENTITY % InlNoAnchor.class
     "%InlStruct.class;
      %InlPhras.class;
      %InlPres.class;
      %InlSpecial.class;
      %InlForm.class;
      %Inline.extra;"
>

<!ENTITY % InlNoAnchor.mix
     "%InlNoAnchor.class;
      %Misc.class;"
>

<!ENTITY % Inline.mix
     "%Inline.class;
      %Misc.class;"
>

<!-- Block Elements ................................. -->

<!ENTITY % Heading.class
     "%h1.qname; | %h2.qname; | %h3.qname;
      | %h4.qname; | %h5.qname; | %h6.qname;"
>
<!ENTITY % List.class  "%ul.qname; | %ol.qname; | %dl.qname;" >
```

```
<!ENTITY % Table.class "| %table.qname;" >

<!ENTITY % Form.class  "| %form.qname;" >

<!ENTITY % BlkStruct.class "%p.qname; | %div.qname;" >

<!ENTITY % BlkPhras.class
     "| %pre.qname; | %blockquote.qname; | %address.qname;"
>

<!ENTITY % BlkPres.class "| %hr.qname;" >

<!ENTITY % BlkSpecial.class
     "%Table.class;
      %Form.class;"
>

<!ENTITY % Block.extra "" >

<!ENTITY % Block.class
     "%BlkStruct.class;
      %BlkPhras.class;
      %BlkPres.class;
      %BlkSpecial.class;
      %Block.extra;"
>

<!ENTITY % Block.mix
     "%Heading.class;
      | %List.class;
      | %Block.class;
      %Misc.class;"
>

<!-- All Content Elements ............................. -->

<!ENTITY % FlowNoTable.mix
     "%Heading.class;
      | %List.class;
      | %BlkStruct.class;
      %BlkPhras.class;
      %BlkPres.class;
      %Form.class;
      %Block.extra;
      | %Inline.class;
      %Misc.class;"
>

<!ENTITY % Flow.mix
     "%Heading.class;
      | %List.class;
      | %Block.class;
      | %Inline.class;
      %Misc.class;"
>

<!-- end of xhtml-print10-model-1.mod -->
```